Attorney Docket: ST999007/1281P

UNITED STATES PATENT APPLICATION

FOR

LAYER VIEWPORT FOR ENHANCED VIEWING IN LAYERED DRAWINGS

Inventors:
Gregory M. Plow
Farrokh E. Pourmirzaie

Sawyer & Associates
2465 E. Bayshore Road, Suite 406
Palo Alto, California 94303

NON-DISRUPTIVE APPLICATION VIEWING AND CAPTURE CAPABILITY

FIELD OF INVENTION

The present invention relates generally to a system and method for the use of an application tool to facilitate the viewing of normally hidden information in a multiple application window environment.

BACKGROUND OF THE INVENTION

A windows type desktop such as IBM OS2 or Microsoft Windows provides the user with substantial flexibility in managing the look of the desktop while interfacing with applications. Functions are provided that enable the desktop user to size application windows, move application windows, overlay application windows and to reduce application windows to icons. An application icon provides only the identification of the deployed application with no other information related to the application visible on the desktop. Figure 1 illustrates an example of a conventional display screen 25 containing desktop icons 15, an application window 20, a minimize button 10, a window reduction/enlargement button 12, a close button 14, an application title bar 16, an application tool bar 18, and scroll bars 22, 24. Application icons 28, are located in the taskbar 26.

Over the years, more and more applications are typically available for activation from the desktop. Generally users are also deploying an increasing number of applications that are concurrently in some state of execution. As this trend continues, even more flexibility for managing the desktop is needed to provide the user with the best productivity from her desktop system. More particularly, when information from two different application windows is needed, the desktop user is forced into one of several methods which can be time consuming and detrimental to her productivity. One such method involves reducing one or more applications windows to an icon such that the underlying application window and information may be viewed. Another method involves selecting the icon representing the application (or selecting a portion of the application window if not completely covered) that contains the needed information as a means to bring that application window to the top of the desktop.

This problem becomes more evident when dealing with overlapping application windows. Figure 2 is illustrative of a situation involving a primary application window 30 and an overlapped application window 40. When there is a need for the simultaneous viewing of information contained in the windows 30, 40, the user must resize the corresponding application windows and arrange them on the desktop so as to make both sets of needed information simultaneously viewable. After viewing the needed information, the user typically resizes the application windows 30, 40 back to their original size to facilitate the easy viewing of each application windows without causing undue eyestrain. When all the user wants is a quick look at some information in one application for comparison or manipulation in the other application, the conventional methodology is a time consuming and annoying procedure.

Productivity is further impacted when the needed information is located in an application window that is not directly beneath the primary application window. If a user is working on a document in the primary application window, he might be interested in a document that is several layers beneath the primary application window. Further, the document of interest may be a completely different application. Typically, the user would have to resize, move, and minimize windows on the desktop screen in order to view multiple documents or applications of interest. This process is time consuming and can cause the user to become distracted.

Another drawback to the conventional methodology relates to a user's inability to quickly move or relocate information from one window to another document. In order to do this, a user would typically have to disrupt the viewing of a primary application window to locate the desired information to cut and paste.

What is needed is a method for viewing the contents of an application window that would otherwise be hidden. This will provide the user with the ability to quickly access needed information without having to undergo the time consuming process of resizing, minimizing or rearranging desktop windows.

SUMMARY OF THE INVENTION

The present invention provides a method, system, and computer readable medium for viewing hidden windows on a display including a plurality of application windows, a portion of at least one application window of the plurality of application windows being obscured. The method, system, and computer readable medium comprise creating a viewport displayed as a topmost application window in response to a user interaction and displaying at least a portion of the at least one application window being obscured in the viewport.

Through the use of the present invention a user can simultaneously view information from multiple applications where one or more application windows are partially or wholly covering needed information in an underlying application window.

Furthermore, the user will also be able to view application windows that are not directly underneath the primary application window, but are in an inactive state. This will provide the user with the ability to quickly access needed information without having to undergo the time consuming process of resizing, minimizing, or rearranging desktop windows.

BRIEF DESCRIPTION OF DRAWINGS

Figure 1 shows a conventional application window configuration.

Figure 2 shows conventional overlapping application windows.

Figure 3 shows a high-level flowchart of the method in accordance with the present invention.

Figure 4 is a diagram of an application viewport in accordance with the present invention.

Figure 5a illustrates an application window.

Figure 5b illustrates an application viewport configuration in accordance with the present invention.

Figure 6 is a second diagram of an application viewport configuration in accordance with the present invention.

Figure 7 is a detailed flowchart of the method in accordance with the embodiment of present invention.

Figure 8 represents an application viewport configuration in accordance with the present invention.

Figure 8a is another application viewport configuration in accordance with the present invention.

Figure 9 represents another embodiment of a method and system in accordance with the present invention.

Figure 10 represents a flowchart of the operation of the capture feature of the method and system in accordance with the present invention.

Figure 11 represents an application configuration illustrative of the capture feature of the method and system in accordance with the present invention.

DETAILED DESCRIPTION OF THE INVENTION

The present invention provides a method and system for viewing previously hidden application windows. The following description is presented to enable one of ordinary skill in the art to make and use the invention and is provided in the context of a patent application and its requirements. Various modifications to the preferred embodiment will be readily apparent to those skilled in the art and the generic principles herein may be applied to other embodiments. Thus, the present invention is not intended to be limited to the embodiments shown but is to be accorded the widest scope consistent with the principles and features described herein.

The present invention allows a user to view information in application windows that are obscured by one or more other application windows. The preferred embodiment allows the user to view information in windows that may or may not be directly beneath the primary application window. This will provide the user with the ability to quickly access needed information without having to undergo the time consuming process of resizing, minimizing, or rearranging desktop windows.

To describe the operation of the method in accordance with the present invention, refer now to Figure 3. Figure 3 is a high-level flowchart of a method in accordance with the present invention. First, an application viewport, displayed as the topmost window, is created in response to a user interaction, via step 50. Next, an application is associated with the viewport in response to a user interaction, via step 55. Next, a portion of the associated application window is displayed in the viewport, via step 60. Note that by default, the application viewport has an initial size and is associated with the primary application window when it is first created. It is also preferable that the application viewport have the ability to view information from different application programs. For example, information from a Microsoft Word application window could be displayed in the viewport as the topmost window, while the primary application window displays information from a Microsoft Excel application.

In a preferred embodiment, the application viewport is created using a desktop tool that can be selected by the user for the creation of a variable sized application viewport. The tool is preferably selected from the Start Menu or a desktop icon. The viewport allows a user to view a portion of a window that is obscured by one or more layers of windows. In addition, the viewport allows a user to view information in an application window different from the primary application window.

Figure 4 represents an application viewport 100 in accordance with the present invention. The title 102 of the application currently being associated with the viewport is postfixed with the word “viewport” and placed in the title bar 108. Also included are a minimize all button 104, a minimize button 105, and a close button 106. Once created, the application viewport can be moved about the desktop or resized and allows a user to view an associated application window even though that application window is otherwise hidden from view by one or more layers of other windows.

However, a situation can arise when the needed information is within a scrollable area of the associated application window but not yet within the application window. To avoid needing to first bring the associated application window that contains the needed information to the top layer of the desktop, then scrolling to the needed information and finally restoring the primary application window to its original position on the desktop prior to resuming use of the application viewport, it is preferable to incorporate a scrolling feature into the viewport. Accordingly, whenever the viewport is dragged off the associated application window boundaries in a horizontal direction, the application associated with the application viewport scrolls in that same horizontal direction, simulating the effect of allowing the application viewport to move unabated to any area within the associated application window. In like fashion, vertical scrolling occurs when the application viewport is dragged off the associated application window boundaries in the vertical direction.

To better understand the scrolling feature, refer to Figure 5a, Figure 5b and Figure 6. Figure 5a illustrates an application window 40' (e.g. Application$_2$- Price List) wherein the "Rocket Printer" listing is beyond the vertical boundary 42 and is therefore not shown. Figure 5b illustrates an application viewport configuration in accordance with the present invention wherein the application viewport 160 is associated with the hidden application window 40' and is located within the primary application window 150. Figure 6 illustrates a second perspective view of the application viewport configuration wherein the application viewport 160' has been dragged to a second position within the primary application window 150 and has reached the associated application window's vertical boundary 42. Accordingly, the application viewport 160' now contains the "Rocket Printer" listing because the associated application window 40' has scrolled in the vertical direction based on the application viewport 160' having been dragged to the associated application window's vertical boundary 42.

To further demonstrate the operation of the method in accordance with the present invention, refer now to Figures 7, 8, and 8a. Figure 7 is a detailed flowchart of the preferred embodiment of the method in accordance with the present invention. Figures 8 and 8a are illustrations of application viewport configurations in accordance with the present invention. Once a user decides to see a previously hidden application window, the user selects the application viewport tool, preferably from the start menu or a desktop icon, via step 200. Next, the application viewport is displayed in the primary application window, via step 202. Initially, the application viewport 100' has a default size and is associated with the primary application window. However, the application viewport can be resized at any time preferably through the use of conventional mouse operations. The user then selects the application icon of interest from the taskbar 222 with a right mouse click, via step 204.

Although the application viewport displayed is initially created with a default size, one of ordinary skill in the art will readily recognize that a variety of methods could be implemented to create and size of the viewport and still remain within the spirit and scope of the present invention. For example, the "click, hold, and drag" method could be utilized. Using this method, a mouse can be utilized to place the cursor at the desired location of the upper left hand corner of the viewport. The mouse button is then depressed and held while the mouse is used to drag the cursor to the desired location of the lower right hand corner of the viewport. The mouse button is then released.

The selection of the application icon of interest is preferably done by right mouse clicking the application icon of interest in the taskbar. However, an alternate method of associating an application window with the application viewport is to depress the “TAB” key while the viewport is active which will cause the various launched applications on the desktop to cycle through the application viewport. The title of the viewport will indicate which application is currently associated with the viewport. Figure 8 shows the viewport 100’ before step 204 where the associated application is the default primary application window 220.

Referring back to Figures 7, the selected application is associated with the application viewport 100’ and the portion of the associated application window beneath the viewport is displayed in the viewport 100’, via step 206. Figure 8a shows the viewport 100’ now associated with the selected application 108.

The user can then size and locate the application viewport 100’ to see relevant portions of the associated application 108, via step 208. Finally, the user can click the close button 106’ to deactivate the application viewport 100’, via step 210.

Figure 9 illustrates another embodiment of the system and method in accordance with the present invention. In the embodiment of Figure 9, the user has the ability to create multiple application viewports 100’’, 100’’’. A related “minimize all” button is preferably located in the title bar of the viewports and allows the user to reduce all application viewports on the desktop to an icon in the taskbar. A subsequent mouse click of the icon, makes all the application viewports visible again. Left mouse clicking on any one of the application viewports makes that viewport the current viewport and makes it eligible to receive a new association.

A further enhancement of the application viewport provides for "capturing" needed information when it becomes visible within the application viewport. This can be accomplished by "right clicking" the mouse within the application viewport. The information displayed within the application viewport is now frozen and constant and the "captured" information may be relocated anywhere by simply moving the viewport to any desired location on the desktop.

To describe the operation of the "capture" feature of this invention refer back to Figure 8a and to Figure 10. Figure 10 is a flowchart of the operation of the capture feature. The application viewport 100' contains a portion of an associated application 108 and is displayed as the topmost window in the primary application window 110. The taskbar 222 preferably contains currently launched application icons.

Referring now to Figures 8a and 10, once a user decides to capture information from a previously hidden application window, the user activates the application viewport tool, via step 300. Next, the application viewport 100' is created with an initial default size in the primary application window as the topmost window, via step 302. The user then selects the application icon of interest from the taskbar 222, via step 304. Next, the selected application 108 is associated with the application viewport 100' and the portion of the associated application window beneath the viewport is displayed in the viewport 100', via step 306. The user then sizes and locates the application viewport 100' to a desired location to find the relevant information of the associated application 108, via step 308. Finally, the user "right clicks" the mouse inside the viewport to create a capture window, via step 310. The "captured" information in the capture window can now be relocated on the desktop to a desired location, via step 312. Figure 11 depicts the "capture" window 112 that has been relocated to a new location on the desktop.

Note that the capture window no longer functions as a viewport in that the captured information within the captured window remains constant as the capture window is moved on the display. Further, the associated application window remains unchanged in response to a capture operation. In addition, information within the capture window is available for conventional cut and paste operations.

This is a nice feature in that it allows the user to "capture" pieces of information scattered at various locations from within hidden window across multiple applications and collect them into one area of the desktop, thereby making available a multitude of reference information while viewing the primary application. The capture window can be immediately restored to its normal state of operation (e.g. viewport) by simply "double clicking" on it.

The system and method in accordance with the present invention allows the user to view the contents of application windows that would otherwise be hidden. Through the use of the present invention a user can simultaneously view information from multiple applications where the primary application window is partially or wholly covering needed information in an underlying window. Furthermore, the user will also be able to view application windows that are not directly underneath the primary application window, but are in an inactive state. This will provide the user with the ability to quickly access needed information without having to undergo the time consuming process of resizing or rearranging desktop window configurations.

Although the present invention has been described in accordance with the embodiments shown, one of ordinary skill in the art will readily recognize that there could be variations to the embodiments and those variations would be within the spirit and scope of the present invention. Accordingly, many modifications may be made by one of ordinary skill in the art without departing from the spirit and scope of the appended claims.